Watcher 개발 문서

조원혁 엔지니어

목차.

1. 화면 템플릿

1-1. default

a. metronic_index.html

기본설명	파일 이름 처럼, 메인 화면을 구성한다. 나머지 다른 html들을 대부분 내포하며, 필요한 스크립트, CSS 파일 등을 import 한다.
특징	로그인 여부 및 접속 URL에 따라 다른 페이지를 contents에 보여준다.
비고	페이지 이름을 넘겨 주어야 한다.

b. html_begin.html

기본설명	html 이 시작하는 영역이며, 우선 필요한 기본 scripts 들을 import 한다.
특징	static 파일 또는 외부 CDN을 이용하여 스크립트들을 가져온다.
비고	메인페이지 로딩시 필요한 스크립트는 여기서 import 하여야 한다.

c. head.html

기본설명	head 태그가 시작되는 파일이며, CSS 파일들을 import 한다.
특징	static 파일 또는 외부 CDN을 이용하여 CSS파일들을 가져온다.
비고	

d. header.html

기본설명	metronic 페이지의 상단 부분 (헤더영역) 을 구성하는 페이지 이다. UTC 및 한국 시간에 대한 스크립트가 내장되어 있다.
특징	
비고	

e. footer.html

기본설명	metronic 페이지의 하단 부분 (풋터영역) 을 구성하는 페이지 이다.
특징	하단 로고를 포함한다.
비고	

f. run_menu.html

기본설명	각 페이지 마다의 마우스 오른쪽 버튼 메뉴를 포함하고 있다.
특징	마우스 오른쪽 버튼 메뉴의 이벤트 연결을 담당 한다.
비고	

g. sidebar.html

기본설명	사이드바를 구성하는 페이지 이다. 각 메뉴마다의 연결을 담당한다.
특징	각 페이지를 Ajax 로 가져오므로, 끊김 현상이 없다.
비고	

h. html_end.html

기본설명	페이지의 마지막에 포함되는 페이지 이다. 기본 페이지에서 필요없는 script들을 여기서 import 해온다. 이를 뒤 영역으로 뺌으로써 페이지의 초기 로딩 속도를 높여 준다.
특징	시간 설정 스크립트가 내포 되어 있다.
비고	

1-2. category

a. list.html

기본설명	카테고리 리스트들을 보여주는 페이지 이다. 테이블 안쪽의 데이터 영역의 스크립트만을 포함하고 있다.
특징	다른 리스트 파일과 달리 스크립트만 내포 하고 있다.
비고	table.html 에서 include 된다.

b. table.html

기본설명	카테고리 테이블을 보여주는 페이지 이다. 테이블에 필요한 이벤트들의 스크립트들을 내포하고 있다.
특징	카테고리 리스트를 직접 html을 텍스트로 렌더링하여 표현한다.
비고	

c. register.html

기본설명	새 카테고리를 등록하는 페이지 이다. 등록에 필요한 이벤트들의 스크립트들을 모두 내포 하고 있다.
특징	모두 ajax로 통신함으로써 끊김 현상이 없다.
비고	

d. page.html

기본설명	table, list 의 html 페이지들을 include 하여 전체 페이지를 렌더링하게 된다.
특징	다른 페이지들을 연결하기 위한 용도로만 사용된다.
비고	

1-3. event

a. list.html

기본설명	이벤트 리스트들을 보여주는 페이지 이다. 테이블 안쪽의 데이터 영역의 페이지 및 스크립트로 구성되어있다.
특징	table.html 에서 include 된다.
비고	

b. table.html

기본설명	이벤트 테이블을 보여주는 페이지 이다. 테이블에 필요한 이벤트들의 스크립트들을 내포하고 있다.
특징	테이블에서 할 수 있는 다양한 이벤트의 스크립트를 모두 포함한다.
비고	

c. register.html

기본설명	새 이벤트를 등록하는 페이지 이다. 등록에 필요한 이벤트들의 스크립트들을 모두 내포 하고 있다.
특징	모두 ajax로 통신함으로써 끊김 현상이 없다.
비고	

1-4. server

a. list.html

기본설명	서버 리스트들을 보여주는 페이지 이다. 테이블 안쪽의 데이터 영역의 페이지 및 스크립트로 구성되어있다.
특징	table.html 에서 include 된다.
비고	

b. table.html

기본설명	서버 테이블을 보여주는 페이지 이다. 테이블에 필요한 이벤트들의 스크립트들을 내포하고 있다.
특징	테이블에서 할 수 있는 다양한 이벤트의 스크립트를 모두 포함한다.
비고	

c. simple_list.html

기본설명	서버 리스트들을 최소화된 컬럼들만 보여주는 페이지 이다. 테이블 안쪽의 데이터 영역의 페이지 및 스크립트로 구성되어있다.
특징	simple_table.html 에서 include 된다.
비고	

d. simple_table.html

기본설명	서버 테이블을 최소화된 컬럼들만 보여주는 페이지 이다. 테이블에 필요한 이벤트들의 스크립트들을 내포하고 있다.
특징	테이블에서 할 수 있는 다양한 이벤트의 스크립트를 모두 포함한다.
비고	

e. register.html

기본설명	새 서버를 등록하는 페이지 이다. 등록에 필요한 이벤트들의 스크립트들을 모두 내포 하고 있다.
특징	모두 ajax로 통신함으로써 끊김 현상이 없다.
비고	

1-5. report

a. list.html

기본설명	리포트 리스트들을 보여주는 페이지 이다. 테이블 안쪽의 데이터 영역의 페이지 및 스크립트로 구성되어있다.
특징	table.html 에서 include 된다.
비고	

b. table.html

기본설명	리포트 테이블을 보여주는 페이지 이다. 테이블에 필요한 이벤트들의 스크립트들을 내포하고 있다.
특징	테이블에서 할 수 있는 다양한 이벤트의 스크립트를 모두 포함한다.
비고	

c. simple_list.html

기본설명	리포트 리스트들을 최소화된 컬럼들만 보여주는 페이지 이다. 테이블 안쪽의 데이터 영역의 페이지 및 스크립트로 구성되어있다.
특징	simple_table.html 에서 include 된다.
비고	

d. simple_table.html

기본설명	서버 테이블을 최소화된 컬럼들만 보여주는 페이지 이다. 테이블에 필요한 이벤트들의 스크립트들을 내포하고 있다.
특징	테이블에서 할 수 있는 다양한 이벤트의 스크립트를 모두 포함한다.
비고	

e. search_report.html

기본설명	리포트를 새로 진행할 수 있는 페이지 이다. 서치에 필요한 시간 (약 1분) 만큼 요청을 날린 후 기다리는 작업이 진행된다.
특징	pdf 미리 보기 등을 할 수 있다.
비고	

f. pdf_form.html

기본설명	리포트를 새로 진행할 수 있는 페이지 이다. 서치에 필요한 시간 (약 1분) 만큼 요청을 날린 후 기다리는 작업이 진행된다.
특징	pdf 미리 보기 등을 할 수 있다.
비고	

1-6. login

a. login.html

기본설명	처음 로그인을 하는 페이지이다. 토큰을 이용하여 보안이 되어 있으며, 세션을 이용한다.
특징	이메일과 패스워드 저장이 가능하다. (세션, 보안)
비고	

b. create_new_user.html

기본설명	회원가입을 하는 페이지이다. 이메일 인증을 통해서 회원가입을 할 수 있도록 설계되었다.
특징	이메일 인증과정을 거친다. 어드민 계정 생성은 마스터 어드민에서 따로 수정해 주어야 한다.
비고	보안코드만 전송 후 입력하면 되므로, 이메일이 아닌 다른 매체로도 수정이 가능하다.

1-7. setup

a. setup.html

기본설명	일반 유저, 어드민 모두가 가능한 환경설정을 담당 하는 페이지이다.
특징	ajax 통신으로 끊김 현상이 없다.
비고	

b. admin_setup.html

기본설명	어드민 모두만 가능한 환경설정을 담당 하는 페이지이다.
특징	ajax 통신으로 끊김 현상이 없다.
비고	로그서버 변경, 전체 업데이트등은 몇 분정도 소요 될 수 있다.

2. 화면 컨트롤러

2-1. urls.py

URL	기본 설명
/main	메인페이지를 호출한다. 로그인 상태가 아닐경우 로그인 화면이 뜨게 된다.
/login	로그인 페이지 호출
/logout	로그아웃 기능
/sign_up	회원가입 기능
/send_certCode	토큰 이메일 전송
/create_new_user	새 유저 생성 페이지 호출
/dashboard	대쉬보드 페이지 호출
/event	이벤트에 대한 RestFul Api를 제공
/event_register	이벤트 등록 페이지 호출
/server	서버에 대한 RestFul Api를 제공
/server_register	서버 등록 페이지 호출
/category	카테고리에 대한 RestFul Api를 제공
/category_options	카테고리 목록을 select 박스 옵션 형태로 제공
/server_options	서버 목록을 select 박스 옵션 형태로 제공
/server_simple_table	최소 서버 테이블 페이지 호출
/report_simple_table	최소 리포트 테이블 페이지 호출
/get_run_button	마우스 오른쪽 버튼 메뉴 제공

URL	기본 설명
/new_server	새 서버 등록 페이지 호출
/new_event	새 이벤트 등록 페이지 호출
/page_server	서버 페이지 호출 (리로딩시)
/page_server_category	서버 카테고리 페이지 호출 (리로딩시)
/page_event	이벤트 페이지 호출 (리로딩시)
/page_event_category	이벤트 카테고리 페이지 호출 (리로딩시)
/page_new_server	새 서버 등록 페이지 호출 (리로딩시)
/page_new_event	새 이벤트 등록 페이지 호출 (리로딩시)
/page_setup	환경 설정 페이지 호출 (리로딩시)
/page_admin_setup	어드민 계정 환경 설정 페이지 호출 (리로딩시)
/attack_start	공격 (패킷 플러딩, http 플러딩) 시작
/attack_stop	공격 중지
/search_start	탐색 (nmap, url check) 시작
/search_stop	탐색 중지
/reboot	서버 재부팅
/shutdown	서버 종료
/delete_all	서버 내부 파일 모두 삭제
/get_events_status	탐색 상태를 반환
/get_servers_status	서버 상태를 반환
/get_servers_system_status	서버의 시스템 상태를 반환
/set_server_status_hold	서버의 상태를 대기중으로 전환
/check_monitoring	각 서버에서 glances 를 실행
/stop_monitoring	각 서버에서 glances 를 중지
/scripted_grafana	스크립트 형태의 그라파나 페이지를 생성
/start_glances	각 서버에서 glances 를 실행
/influxdb_start	로그 서버에서 *influxdb* 실행
/mysqld_start	통제 서버에서 mysql 실행
/grafana_start	통제 서버에서 grafana 실행

URL	기본 설명
/search_batch_start	이벤트의 배치 작업을 시작
/search_report	리포트 생성 페이지 호출
/new_report	리포트 생성 페이지 호출 (리로딩시)
/search	탐색(리포트)에 대한 RestFul Api를 제공
/get_event_servers	한 이벤트에 속한 서버의 리스트를 반환
/get_event_target_servers	한 이벤트에 속한 대상 서버의 리스트를 반환
/create_report	리포트 생성
/search_report_result	리포트 결과 출력
/search_pdf	pdf 형태로 변형할 웹 페이지 형태의 보고서 반환
/create_pdf	웹 페이지 형태의 보고서를 pdf로 변환하여 파일로 저장
/setup	환경 설정 페이지 호출
/admin_setup	어드민용 환경 설정 페이지 호출
/change_log_server	전체 서버에 대한 로그 서버를 재정의
/do_bash_all	모든 서버에 bash 명령어를 수행하도록 함

2-2. views.py
- urls.py 에 있는 url 과 동일한 명의 메소드들을 포함하고 있으며, 위 표에 정리된 내용을 직접 수행하는 코드들이 모여 있는 파이썬 파일.

3. 모델

3-1. models.py

a. User

class User	기본 설명
userSrl = models.AutoField(primary_key=True)	user 테이블의 auto increment PK
password = models.CharField(max_length=128)	비밀번호

class User	기본 설명
fullname = models.CharField(max_length=50, blank=True)	이름
email = models.CharField(max_length=50, blank=True, unique=True)	메일
address = models.CharField(max_length=50, blank=True)	주소
phone = models.CharField(max_length=50, blank=True)	폰번호
birthYear = models.CharField(max_length=10, blank=True)	생일
sex = models.CharField(max_length=5, blank=True)	성별
imageUrl = models.FileField(null=True, upload_to='image/%Y/%m/%d',blank=True)	유저 프로필 이미지
certificateCode = models.CharField(max_length=10, blank=True)	유저 인증번호
isCertificated = models.BooleanField(default=False)	인증 완료 여부
isDisabled = models.BooleanField(default=False)	계정 정지 여부
isAdmin = models.BooleanField(default=False)	어드민 여부
regTime = models.DateTimeField(auto_now_add=True)	등록 시간
updTime = models.DateTimeField(auto_now=True)	마지막 수정 시간
grafanaRefresh = models.IntegerField(default=30)	그라파나 주기 설정 시간
searchUserSrl = models.CharField(max_length=5000, default='')	어드민일 경우, 다른 계정 컨트롤을 위한 필드 (현재 사용 안함)
logServer = models.IntegerField(default=0)	등록한 로그 서버 srl 정보
isSearchedByAdmin = models.BooleanField(default=False)	해당 계정이 어드민으로 부터 컨트롤 되고 있는지를 나타내는 필드
lastLoginTime = models.DateTimeField(auto_now=True)	마지막 로그인 시간

b. Category

class Category	기본 설명
categorySrl = models.AutoField(primary_key=True)	카테고리 테이블 auto increment PK
parentCategorySrl = models.IntegerField(default=0)	카테고리의 부모 Srl
type = models.CharField(max_length=100, choices=category_type, default='server')	카테고리의 종류 (현재 사용 안함)
name = models.CharField(max_length=100)	카테고리 이름
description = models.CharField(max_length=1000)	카테고리 설명
priority = models.IntegerField(default=100)	우선순위
isEssential = models.BooleanField(default=False)	필수 여부
regTime = models.DateTimeField(auto_now_add=True)	등록 시간
updTime = models.DateTimeField(auto_now=True)	마지막 수정 시간

c. Server

class Server	기본 설명
serverSrl = models.AutoField(primary_key=True)	서버 테이블 auto increment PK
id = models.CharField(max_length=100, default='')	서버 아이디
name = models.CharField(max_length=100)	서버이름
ip = models.CharField(max_length=100)	서버 IP
port = models.IntegerField(default=80)	서버 port
type = models.CharField(max_length=50, choices=server_type)	서버 종류
user = models.ForeignKey(User)	서버가 속한 유저
category = models.ManyToManyField(Category, through='ServerCategory')	서버가 속한 카테고리
runCount = models.IntegerField(default=0)	서버가 현재 까지 수행한 작업 횟 수

class Server	기본 설명
status = models.CharField(max_length=50, choices=status, default='hold')	서버 상태
system_status = models.CharField(max_length=20, choices=system_status, default='off')	서버 시스템 상태
monitoring_status = models.CharField(max_length=20, choices=monitoring_status, default='hold')	서버의 모니터링 수행 여부
regTime = models.DateTimeField(auto_now_add=True)	등록 시간
updTime = models.DateTimeField(auto_now=True)	마지막 수정 시간
check_url_pid = models.CharField(max_length=100, blank=True, null=True)	서버에서 수행되고 있는 check_url 의 pid 임시 저장소 (현재 사용안함)
nmap_pid = models.CharField(max_length=1000, blank=True, null=True)	서버에서 수행되고 있는 namp 의 pid 임시 저장소 (현재 사용안함)

d. Event

class Event	기본 설명
eventSrl = models.AutoField(primary_key=True)	이벤트 테이블의 auto increment PK
id = models.CharField(max_length=100, default='')	이벤트 아이디
type = models.CharField(max_length=50, choices=event_type)	이벤트 종류
name = models.CharField(max_length=100)	이벤트 이름
description = models.CharField(max_length=1000)	이벤트 설명
workServer = models.ManyToManyField(Server, related_name='workServer')	작업 수행 서버
targetServer = models.ManyToManyField(Server, related_name='targetServer')	작업 대상 서버

class Event	기본 설명
user = models.ForeignKey(User)	이벤트를 소유한 유저
category = models.ManyToManyField(Category, through='EventCategory')	이벤트가 속한 카테고리
runCount = models.IntegerField(default=0)	이벤트 수행 횟수
status = models.CharField(max_length=50, choices=status, default='hold')	이벤트 상태
startTime = models.DateTimeField(default=timezone.now())	작업 시작 시간
endTime = models.DateTimeField(default=timezone.now())	작업 종료 시간
attackMethod = models.CharField(max_length=200, default="")	공격 방법
regTime = models.DateTimeField(auto_now_add=True)	등록 시간
updTime = models.DateTimeField(auto_now=True)	마지막 수정 시간
pid = models.CharField(max_length=1000, blank=True, null=True, default='')	작업의 pid 임시 저장소 (현재 사용 안함)
check_url_pid = models.CharField(max_length=1000, blank=True, null=True, default='')	check_url_pid 작업의 pid 임시 저장소 (현재 사용 안함)
nmap_pid = models.CharField(max_length=1000, blank=True, null=True, default='')	nmap_pid 작업의 pid 임시 저장소 (현재 사용 안함)
job = models.CharField(max_length=100, blank=True, null=True)	작업 이름
searchLog = models.TextField()	탐색 로그
pps = models.IntegerField(default=1)	초당 공격 횟 수
attackMinute = models.IntegerField(default=1)	공격 지속 시간

e. Search

class Search	기본 설명
searchSrl = models.AutoField(primary_key=True)	탐색 테이블의 auto increment PK
id = models.CharField(max_length=200, blank=True, null=True)	탐색 아이디
report = models.TextField(default='')	보고서 내용
targetServer = models.ForeignKey(Server, related_name='searchTarget')	대상 서버
searchServer = models.ForeignKey(Server, related_name='search')	탐색 수행 서버
status = models.CharField(max_length=50, choices=status_, default='stop')	탐색 상태
startTime = models.DateTimeField(auto_now_add=True)	탐색 시작 시간
endTime = models.DateTimeField(auto_now=True)	탐색 종료 시간
regTime = models.DateTimeField(auto_now_add=True)	등록 시간
updTime = models.DateTimeField(auto_now=True)	마지막 수정 시간
user = models.ForeignKey(User)	탐색을 소유한 유저
portscan = models.TextField(default='')	포트스캔 결과
route = models.TextField(default='')	루트 탐색 결과
check_url = models.TextField(default='')	check url 결과
isDone = models.BooleanField(default=False)	탐색 종료 여부

4. 기타 APIs

4-1. attack_start, attack_stop
- 공격 시작 및 중지하는 API

방식	변수	설명	리턴
GET	srls	이벤트 srl 들을 srls 변수에 리스트 형태로 담아서 보냄	msg : 공격 시작 메시지

4-2. search_start, search_stop
- 탐색 시작 및 중지하는 API

방식	변수	설명	리턴
GET	srls	이벤트 srl 들을 srls 변수에 리스트 형태로 전송	msg : 탐색 시작 메시지

4-3. create_report

방식	변수	설명	리턴
GET	target, worker, id	대상 서버 srl, 수행 서버 srl, 현재 시작으로 된 id	srl : 새로 등록된 탐색 모델의 srl 값

4-4. create_pdf

방식	변수	설명	리턴
GET	searchSrl	create_report를 통해 새로 생성된 탐색 모델의 srl을 전송	빈 json 객체

4-5. reboot, shutdown

방식	변수	설명	리턴
GET	srls	재부팅 혹은 종료 시킬 서버들의 srl 리스트	msg : 성공 메시지

4-6. start_glances, influxdb_start, mysqld_start, grafana_start, search_batch_start

방식	변수	설명	리턴
GET	srls	작업 수행할 서버들의 srl 리스트	빈 json 객체

4-7. change_log_server

방식	변수	설명	리턴
GET	logserverSrl	로그 서버의 등록된 srl	msg : 성공 메시지

4-8. do_bash_all

방식	변수	설명	리턴
GET	cmd	전체 서버에서 수행할 bash 명령어	msg : 성공 메시지

5. Scripts

5-1. upload_to_git.sh, download_from_git.sh

기본 설명	비고
깃 으로 부터 다운 받거나 업로드 할때 사용	

5-2. update_db.sh

기본 설명	비고
데이터 베이스를 업데이트 할 때 사용	통제 서버에서 해주어야 실제 서비스 데이터베이스가 업데이트 됨

5-3. create_super_user.sh

기본 설명	비고
django의 어드민 유저를 생성할 때 사용	통제 서버에서 해주어야 실제 서비스 마스터 어드민 계정이 생성됨 (기본 계정 : whcho888@gmail.com // ql135cjs)

5-4. install-hydra.sh

기본 설명	비고
전체 설치를 할 때 이용하는 인스톨 파일	Cent OS 6 버젼용

5-5. set_api_password.sh

기본 설명	비고
auth api login을 위한 비밀번호 설정	모든 서버에서 설치되어야 함

5-6. search_start.sh

기본 설명	비고
공격, 탐색, Bash API 등을 모두 실행	모든 서버에서 실행 되어야 함

5-5. glances_start.sh, glances_run.sh, glances.conf

기본 설명	비고
glances 를 실행 하는 파일 묶음. glances.conf는 설정사항이 저장되어 있고, glances_run은 이 설정 파일을 읽어 glances를 수행. 마지막으로 glances_start는 glances_run을 새로운 tmux 세션에서 실행 시킴 갱신 주기를 초단위로 argument 로 받음	사용법 : ./glances_start.sh 10 (glances_run.sh 과 glances.conf 는 glances_start.sh 안에서 모두 실행 및 참고 되어 짐)

5-6. runserver.sh

기본 설명	비고
메트로닉의 웹서버를 키는 스크립트	tmux 세션 안에서 키고 접속을 닫기를 권장

5-7. grafana_start.sh, mysqld_start.sh

기본 설명	비고
그라파나 웹서버와 mysql DB 서버를 키는 스크립트	통제 서버에서 실행 함

6. static

설명	특이사항
프로젝트에 필요한 static 파일들을 모아놓은 저장 파일 일반적인 웹서버의 static 파일과 구조 동일 메트로닉 자체의 static 파일들과 커스텀하게 제작된 이미지, script, css 파일 들이 모두 저장	메트로닉 기본 layout들이 모두 저장되어 있음